Machine Learning Channel Estimation of Wireless Communication Signals

Lucy Song[2], Jayden Booth[1], Ahmed Ewaisha[1], Andreas Spanias[1], Cihan Tepedelenlioglu[1]
REU: Sensor, Signal, and Information Processing (SenSIP) Center
[1] School of Electrical, Computer, and Energy Engineering, Arizona State University
[2] School of Computing, Informatics, and Decision Systems Engineering, Arizona State University

***Abstract*—In this paper, machine learning is implemented for channel estimation. While channels can be individually estimated, it may become inefficient and difficult in large multiple-input-multiple-output (MIMO) systems. Thus, for our work, some channels are estimated based on the existing correlation between MIMO channels, while the rest are inferred using a machine learning algorithm. Specifically, this is executed with a simple, yet effective, system consisting of the USRP and LimeSDR-USB communication kits for signal transmission and receiving, respectively. The machine learning algorithm is developed in MATLAB, and is evaluated in the lab to have an accuracy of 37%.**

Keywords—machine learning, LimeSDR, USRP, channel estimation, cognitive radio, wireless, MIMO, artificial intelligence

I. Introduction

Wireless communication has grown considerably since the First Generation (1G) systems of the 1980s. In order to keep up with the growth of data traffic, it is becoming increasingly important to not only develop more efficient technologies that are perceptive and reconfigurable, but also create applications of artificial intelligence (AI) that generate more accurate decision-making and signal classification [1, 2, 4]. Unlike civil applications of wireless communication in which specific transmission schemes or standards are known and assigned to both the transmitted (Tx) and received (Rx), in military applications the standards are generally unknown and thus classification techniques are important and necessary in decoding signals. Without first classifying them, they remain undecodable and useless.

Currently, machine learning, a type of AI, is being used to analyze patterns in wireless communication, including spectrum performance and radio configurations, to improve and optimize the quality of service (QoS) of transmissions [1, 2]. The use of multiple antennas in multiple-input-multiple-output (MIMO) systems are also beneficial in increasing the transmission rate or reducing the bit error rate [3]. A published research has been working closely with the feature recognition (FB) method of machine learning, a Wireless Signal Classification (WSC) algorithm, and higher-order cumulants (HOCs) of received signals in Automatic Modulation Classification (AMC) to distinguish between a variety of modulations [4].

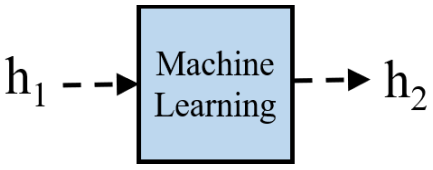

Figure 1. Processing h_1 channel estimation values with machine learning algorithm to predict h_2 channel values.

The FB method is only one of two different AMC algorithms. The other is the likelihood-based (LB) method, but it is regarded as more complex and less efficient because it requires more prior knowledge of the transmitted signal [4, 5]; this is rather impractical, especially for military applications that interact with signals featuring unknown characteristics. The FB method requires less prior knowledge, and relies more on analyzing the features present in the received signal to determine the modulation type [5].

In this work, we conduct and predict channel gain estimation using communication kits and a MATLAB machine learning algorithm. In this setup, we transmit and receive the signals using a USRP radio and a LimeSDR-USB radio featuring MIMO antennas, respectively. Then, we estimate the channel gain between the transmitter (Tx) and receiver (Rx) of a single-input-multiple-output (SIMO) system. A machine learning algorithm in MATLAB is trained using channel estimation values from the SIMO system. As shown in Fig. 1, we aim to pass gain estimations (h_1 vector) of one Rx antenna through a machine learning algorithm to predict channel estimations of successive Rx antennas (h_2 …) of the same radio.

II. Background

A. Channel Estimation with SIMO System

For this work, we chose to simplify our goal to estimating the signal power using normalized transmitted signal power

$$|h|^2 = a^2 + b^2, \quad (1)$$

instead of the complex gain

$$h = a + j*b, \quad (2)$$

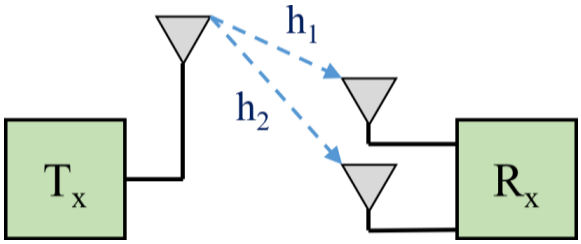

Figure 2. The channel estimation of a single-input-multiple-output (SIMO) system.

of channels. In both (1) and (2), "h" refers to channel gain, and "a" and "b" refer to in-phase and quadrature values, respectively. Determining the power is sufficient enough to reveal the channel's data transmission rate, a value crucial to wireless communication systems.

A SIMO system was specifically chosen for our channel estimations. As Fig. 2 shows, a signal is sent from one antenna and received by two antennas. Based on the difference in location and spacing of the Rx antennas, they receive slightly different versions of the same signal. Determining the difference in gain (h_1 and h_2) is critical for calibration of the wireless communication system.

B. Software Defined Radio (SDR)

We used a Software Defined Radio (SDR) with integrated hardware and software components for more flexible radio configurations [6]. Signal processing features such as FPGAs and DSPs are no longer limited to only hardware implementation.

C. LimeSDR-USB Radio

The specific model used in our work is the LimeSDR-USB. Despite its relative low cost, it features a MIMO, 4-channel, 10-RF connectors system [7]. It has four broadband Tx ports, and six Rx ports (low band <1.5GHz, high band >1.5GHz, and wide/broad band 100kHz—3.8GHz). While the wide band ports are able to detect an extensive frequency range, this capability poses some compromises in performance when compared to the more specifically high or low band ports. Another advantage with the LimeSDR lies in its capability to transmit and receive at the same time—something other software defined radios such as the USRP cannot do. Even though the LimeSDR does have the MIMO capability, the user may also choose SISO instead based on the number of channels desired.

D. LimeSuiteGUI

The LimeSuiteGUI offers a user interface for controlling LimeSDR radio settings. Unlike other similar applications such as GNU Radio, the LimeSuiteGUI was constructed to maximize upon all of LimeSDR functions.

One of the main GUI features used in our work was the FFT Viewer. It shows three graphs, two of which depict IQ sample information and the other an FFT graph. The visible bandwidth of the FFT can be altered accordingly, and for the purposes of our work, we maintained a bandwidth of 20MHz. This is where samples of IQ data were obtained.

E. USRP Radio (B200 Model)

The URSP radio offers more advanced features than the LimeSDR-USB. Additionally, it offers two antennas (1 Rx, 1 Tx), and only one can be used at a time. As shown in Fig. 3, the main purpose of the USRP in this work was for signal transmission. This USRP's frequency range is 70MHz—6GHz [8].

F. GNU Radio Companion

Similar to the LimeSuiteGUI, the GNU Radio is also an application that can be paired with software radios [9]. Dissimilarly, however, it uses block diagrams that correspond to various radio settings. In this work, the GNU Radio was used to control the USRP and transmit a pilot signal.

G. MATLAB Programming

The "classify" command:

$$class = classify(sample, training, group) \qquad (3)$$

was implemented to create a machine learning algorithm for channel estimation [10]. By default, this function conducts supervised learning and linear discriminant analysis. "Class" refers to a testing output of classified data based on the "sample" data set; "sample" refers to a testing input data set that needs to be classified; "training" refers to a training input data set; and "group" refers to the training output data set that holds the classification of the training input data. Correspondingly, the "training" and "group", as well as the "sample" and "class," must have the same row dimensions. There also must be the same number of columns in "sample" and "training".

H. MyriadRF Website, Wiki, & Discourse

The MyriadRF website contains the largest amount of resources for understanding the LimeSDR's functions that cannot be found anywhere else on the Internet. It contains many sets of documentation for background information and radio calibration. Users can troubleshoot their RF issues on the Discourse page and receive help from experts.

Figure 3. The USRP B200 Radio, with only one antenna attached at the Tx port.

III. Methods

A. Channel Estimation Task

A pilot signal was generated in MATLAB, saved as a WAV file, and then uploaded to GNU Radio to be transmitted by the USRP at the frequency of 2.4GHz. The Tx signal magnitude was normalized to the value of 1. Accordingly, in order to receive this signal, the LimeSDR's Rx was tuned to 2.4GHz. The signal reception can be confirmed with the FFT Viewer of LimeSuiteGUI (Fig. 4). In total, 16384 IQ samples of this transmission were collected and stored in a text (.txt) file. Depending on how many channels were in use at the time (1 or 2), this file may store either a two- or four-column vector, with each column denoting I or Q values. For our channel estimation task, we sought to analyze two channels, so our sample file consisted of a 4-column vector.

These samples of the pilot signal were processed through MATLAB. The first two columns of the sample file correspond to channel A, and were extracted into a new 2-column vector h_1; the last two columns correspond to channel B, and these were extracted, transformed using (2) to find the power of each sample, and saved in a new 1-column vector h_2. For both channels A and B, since the Tx signal magnitude was maintained at a value of 1, the gains (h_1 and h_2) are simply the value of the Rx signal magnitudes.

The 16384 samples were divided into 2 groups: 2000 samples for the training data, and the rest (14384 samples) for the testing data. Based on a threshold T, each of the 2000 h_2 samples were classified as either +1 if > T or -1 if < T, and these +1/-1 values were stored in a new 1-column vector. T is determined by averaging the power of all 16384 h_2 samples. When applying (3), the "sample" is now the 14384 h_1 testing vector, "training" the 2000 h_1 training vector, and "group" the classified 2000 h_2 training vector; accordingly, "class" would contain +1/-1 classified values in a 14384-row by 1-column vector.

B. Hardware Assembly and Calibration

The LimeSDR communication kit was set up as shown in Fig. 5. Four Linx-brand antennas were used for the MIMO

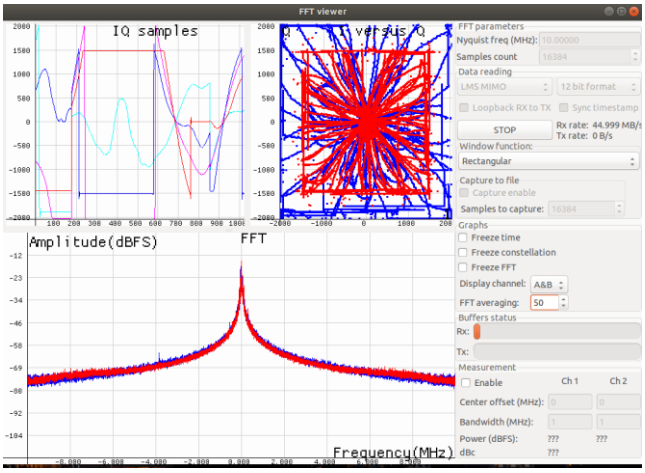

Figure 4. The pilot signal received by the LimeSDR (high band Rx), as shown on the LimeSuiteGUI FFT Viewer.

Figure 5. The assembled LimeSDR communication kit.

system. Two were connected to the broadband Tx1_1 and Tx2_1 ports, while the others were connected to the high band Rx1_H and Rx2_H. Since the board can become very hot during usage, a fan was also attached as a cooling mechanism that automatically turns on and off when the board reaches certain temperatures (On: ~55°C; Off: ~45°C) [11].

The LimeSDR was then connected to a computer and calibrated through the LimeSuiteGUI. We tested the radio's loopback function and the Tx and Rx individually to ensure the Tx and Rx channels were operative. If more than one antenna is used for Tx and/or Rx, each antenna is dedicated its own channel (A or B), which can be seen on the FFT Viewer.

IV. Results, Discussion, & Future Work

Initial results show that there is some correlation between h_1 and h_2 based on an analysis of their correlation coefficient. Its value is 0.2-0.5, and since any coefficient value greater than 0 is indicative of existing correlation, this demonstrates the possibility of using machine learning to infer gain values from one known value.

It was also observed that when switching between the Rx high band and wide band ports, the compromises in performance of the wide band (as noted in MyriadRF documentation) became more pronounced. As Fig. 6 depicts, the signal received through the high band is at least 1250 times stronger. Thus, in future works, unless it is necessary to detect a wide range of frequencies, the more specific high or low bands will be used instead.

In the machine learning process, inaccuracies in classification may still occur despite implementing supervised learning. For the channel estimation task, some h_2 may be misclassified as, for example, -1 when their value is in fact greater than the threshold T. Thus, "accuracy" in this context refers to the number of correctly classified h_2 values out of the total number of h_2. The accuracy of the machine learning algorithm we developed in MATLAB is 37.1872%. The major limitation is due to noise present in Rx samples. In the future, the accuracy could be improved through sample averaging.

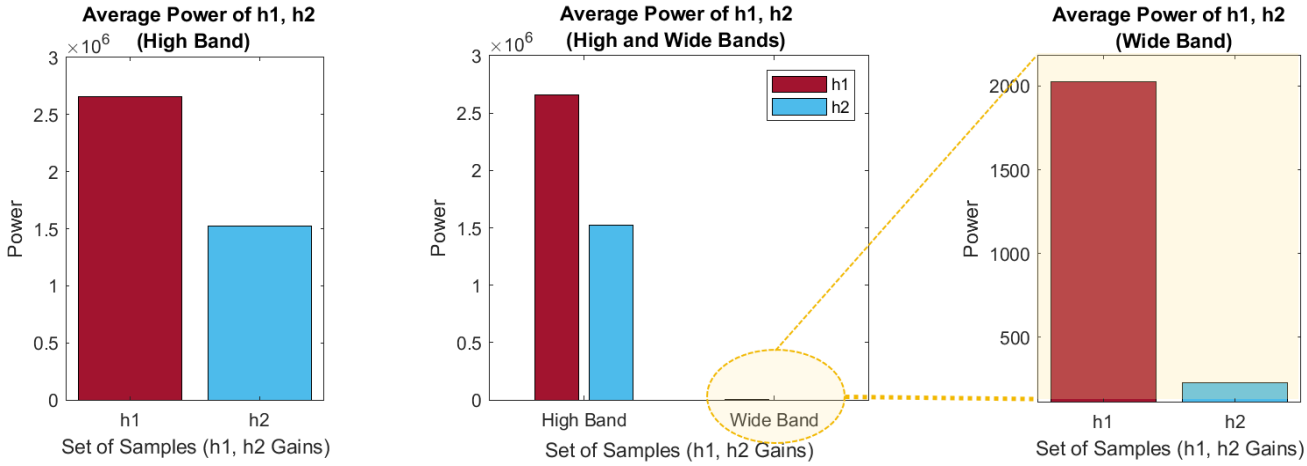

Figure 6. A comparison of average power of h_1, h_2 gains between Rx high band and wide band. The signal detected using the high band ports is clearly more powerful than the wide band. The wide band bars in the middle graph are nearly non-existent due to lower signal power, but the graph on the right provides a closer look at wide band power values.

V. Conclusion

In this paper, we demonstrated that using machine learning to infer gain values is a plausible method, as reaffirmed by an analysis of the h_1 and h_2 correlation coefficient. We implemented a relatively simple, yet effective, setup of a SIMO system, consisting of the LimeSDR and USRP hardware and LimeSuiteGUI and GNU Radio software, to conduct channel estimation.

Implementing machine learning to calculate channel estimation is important because while the gain of each antenna can be estimated individually, it is rather inefficient and difficult in a system with a large number of antennas. Thus, it would be more beneficial to use one or more channel estimates to deduce others in the same RF system.

VI. Acknowledgment

This research is sponsored in part by the NSF CISE Award number 1659871 granted to the Sensor, Signal and Information Processing (SenSIP) Devices and Algorithms REU Site.

VII. References

[1] A. Basu and B. Bhattacharyya, "A Study on the Integration of Machine Learning in Wireless Communication," *2018 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, 2018, pp. 0974-0978.doi: 10.1109/ICCSP.2018.8524369

[2] X. Zhou, M. Sun, G. Y. Li and B. Fred Juang, "Intelligent wireless communications enabled by cognitive radio and machine learning," in *China Communications*, vol. 15, no. 12, pp. 16-48, Dec. 2018.

[3] M. Pappa, C. Ramesh and M. N. Kumar, "Performance comparison of massive MIMO and conventional MIMO using channel parameters," *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, Chennai, 2017, pp.1808-1812. doi: 10.1109/WiSPNET.2017.8300073

[4] Y. Zhang, J. Wang, G. Wu and Q. Tang, "Wireless Signal Classification Based on High-Order Cumulants and Machine Learning," *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, Chongqing, China, 2018, pp. 246-250. doi: 10.1109/IICSPI.2018.8690352

[5] M. S. Mühlhaus, M. Öner, O. A. Dobre, H. U. Jäkel and F. K. Jondral, "A novel algorithm for MIMO signal classification using higher-order cumulants," *2013 IEEE Radio and Wireless Symposium*, Austin, TX, 2013, pp. 7-9. doi: 10.1109/RWS.2013.6486623

[6] A. C. Tribble, "The software defined radio: Fact and fiction," *2008 IEEE Radio and Wireless Symposium*, Orlando, FL, 2008, pp. 5-8. doi: 10.1109/RWS.2008.4463414

[7] K. Woodward, "MyriadRF," *Myriad RF*, 17-Jul-2017. [Online]. Available: https://myriadrf.org/news/limesdr-made-simple-part-1/. [Accessed: 12-Jul-2019].

[8] Ettus Research, "USRP B200 USB Software Defined Radio (SDR)," *Ettus Research*. [Online]. Available: https://www.ettus.com/all-products/ub200-kit/. [Accessed: 12-Jul-2019].

[9] "About GNU Radio · GNU Radio," *GNU Radio*. [Online]. Available: https://www.gnuradio.org/about/. [Accessed: 12-Jul-2019].

[10] "classify," *Discriminant analysis - MATLAB*. [Online]. Available: https://www.mathworks.com/help/stats/classify.html. [Accessed: 12-Jul-2019].

[11] "LimeSDR-USB hardware description," *MyriadRF*. [Online]. Available: https://wiki.myriadrf.org/LimeSDR-USB_hardware_description. [Accessed: 12-Jul-2019].